***Software Engineering***

***Software Requirements Specification***

***(SRS) Document***

**Progress**

**02/14/2023**

**V1**

**By: Dallin Pierce, James Copestake, Devin Holcomb, Gary Li,**

**I have abided by the UNCG Academic Integrity Policy on this Assignment**

**Table of Contents**

[1. Introduction 3](#_Toc126197756)

[1.1. Purpose 3](#_Toc126197757)

[1.2. Document Conventions 3](#_Toc126197758)

[1.3. Definitions, Acronyms, and Abbreviations 3](#_Toc126197759)

[1.4. Intended Audience 4](#_Toc126197760)

[1.5. Project Scope 4](#_Toc126197761)

[1.6. Technology Challenges 4](#_Toc126197762)

[1.7. References 4](#_Toc126197763)

[2. General Description 4](#_Toc126197764)

[2.1. Product Perspective 4](#_Toc126197765)

[2.2. Product Features 4](#_Toc126197766)

[2.3. User Class and Characteristics 5](#_Toc126197767)

[2.4. Operating Environment 5](#_Toc126197768)

[2.5. Constraints 5](#_Toc126197769)

[2.6. Assumptions and Dependencies 5](#_Toc126197770)

[3. Functional Requirements 5](#_Toc126197771)

[3.1. Primary 5](#_Toc126197772)

[3.2. Secondary 5](#_Toc126197773)

[4. Technical Requirements 6](#_Toc126197774)

[4.1. Operating System and Compatibility 6](#_Toc126197779)

[4.2. Interface Requirements 6](#_Toc126197780)

[4.2.1. User Interfaces 6](#_Toc126197781)

[4.2.2. Hardware Interfaces 6](#_Toc126197782)

[4.2.3. Communications Interfaces 6](#_Toc126197783)

[4.2.4. Software Interfaces 6](#_Toc126197784)

[5. Non-Functional Requirements 6](#_Toc126197785)

[5.1. Performance Requirements 6](#_Toc126197786)

[5.2. Safety Requirements 7](#_Toc126197787)

[5.3. Security Requirements 7](#_Toc126197788)

[5.4. Software Quality Attributes 7](#_Toc126197789)

[5.4.1. Availability 7](#_Toc126197790)

[5.4.2. Correctness 7](#_Toc126197791)

[5.4.3. Maintainability 7](#_Toc126197792)

[5.4.4. Reusability 7](#_Toc126197793)

[5.4.5. Portability 7](#_Toc126197794)

[5.5. Process Requirements 7](#_Toc126197795)

[5.5.1. Development Process Used 7](#_Toc126197796)

[5.5.2. Time Constraints 7](#_Toc126197797)

[5.5.3. Cost and Delivery Date 7](#_Toc126197798)

[5.6. Other Requirements 7](#_Toc126197799)

[5.7. Use-Case Model Diagram 8](#_Toc126197800)

[5.8. Use-Case Model Descriptions 8](#_Toc126197801)

[5.8.1. Actor: Actor Name (Responsible Team Member) 8](#_Toc126197802)

[5.8.2. Actor: Actor Name (Responsible Team Member) 8](#_Toc126197803)

[5.8.3. Actor: Actor Name (Responsible Team Member) 8](#_Toc126197804)

[5.9. Use-Case Model Scenarios 8](#_Toc126197805)

[5.9.1. Actor: Actor Name (Responsible Team Member) 8](#_Toc126197806)

[5.9.2. Actor: Actor Name (Responsible Team Member) 9](#_Toc126197807)

[5.9.3. Actor: Actor Name (Responsible Team Member) 9](#_Toc126197808)

[6. Design Documents 9](#_Toc126197809)

[6.1. Software Architecture 9](#_Toc126197810)

[6.2. High-Level Database Schema 9](#_Toc126197811)

[6.3. Software Design 9](#_Toc126197812)

[6.3.1. State Machine Diagram: Actor Name (Responsible Team Member) 9](#_Toc126197813)

[6.3.2. State Machine Diagram: Actor Name (Responsible Team Member) 9](#_Toc126197814)

[6.3.3. State Machine Diagram: Actor Name (Responsible Team Member) 9](#_Toc126197815)

[6.4. UML Class Diagram 9](#_Toc126197816)

[7. Scenario 10](#_Toc126197817)

[7.1. Brief Written Scenario with Screenshots 10](#_Toc126197818)

# Introduction

## Purpose

[The goal of your project and the objectives it wishes to accomplish]

The goal of Progress is to allow gym users of all abilities to have a community which is easily accessible that acts as a catalyst to help everyone to progress in their fitness journey. Users can talk to one another; ask whatever questions they may have and use the knowledge of others and what they do to adjust their routine to reach the goals that they have set themselves quicker and more efficiently.

Another big goal of Progress is to encourage beginners who may have been to the gym previously but been intimidated or people who have never been before to go to the gym and by having this software available it will show these users to speak to other people who go to the gym thus allowing them to have a easier time when they begin the gym as they have been able to ask questions and get to know people prior to going.

## Document Conventions

[Full description of the main objectives of this document in the context of your project.

Here’s how you should begin this section:

“The purpose of this Software Requirements Document (SRD) is to...”

“In it, we will . . ., . . ., and . . ..”]

The purpose of this Software Requirements Document (SRD) is to explain and portray the features and functions that will be within the gym social media platform “Progress”. In the application we aim to include the following features and functions:

* Private messaging between users of the platform
* Users creating different posts, be these questions, videos
* Users can react to other users posts and comment on them
* Certain users can publicize their services (for example personal training appointments)
* A level system for the users (beginner, intermediate, advanced) to distinguish what roles and features they have available to them on the application.

## Definitions, Acronyms, and Abbreviations

[Include any specialized terminology dictated by the application area or the product area.

For example:]

|  |  |
| --- | --- |
| Java | A programming language originally developed by James Gosling at Sun Microsystems. We will be using this language to build the Restaurant Manager. |
| MySQL | Open-source relational database management system. |
| .HTML | Hypertext Markup Language. This is the code that will be used to structure and design the web application and its content. |
| SpringBoot | An open-source Java-based framework used to create a micro-Service. This will be used to create and run our application. |
| MVC | Model-View-Controller. This is the architectural pattern that will be used to implement our system. |
| Spring Web | Will be used to build our web application by using Spring MVC. This is one of the dependencies of our system. |
| Thymeleaf | A modern server-side Java template engine for our web environment. This is one of the dependencies of our system. |
| NetBeans | An integrated development environment (IDE) for Java. This is where our system will be created. |
| API | Application Programming Interface. This will be used to implement a function within the software where the current date and time is displayed on the homepage. |

## Intended Audience

[Describe which part of the SRS document is intended for which reader. Include a list of all stakeholders of the project, developers, project managers, and users for better clarity.]

Stakeholders

* Gym Users
* Gym Owners
* Personal Trainers
* UI/ UX Designers
* Software Engineers
* Project Manager

SRS Document and Readers

1. Introduction – UI/UX Designers, Software Engineers & Project Manager
2. General Description - UI/UX Designers, Software Engineers & Project Manager
3. Functional Requirements – UI/UX Designers & Software Engineers
4. Technical Requirements – UI/UX Designers & Software Engineers
5. Non-Functional Requirements - Software Engineers
6. Design Documents – Software Engineers
7. Scenario - UI/UX Designers & Software Engineers

## Project Scope

[Specify how the software goals align with the overall business goals and outline the benefits of the project to business.]

The goal of the software is to provide a new environment to users that can use it as a source of information for everything related to fitness and personal development. The environment will contain an easy-to-use UI with UX at the forefront and simple to use features. This aligns with the overall business goals of a social media platform as these platforms would fail if the purposes of the intended software are difficult to use with a confusing design therefore resulting in a rapid decline in users and no to minimal activity on the software.

The benefits of the project to business include:

* In helping alleviate the stress new gym goers face when thinking about starting the gym, this software should hopefully increase the intake gyms due to removing this fear.
* Not only will this software help people get into the gym but will also expose them to everything that goes hand in hand with the gym such as dieting, supplements and other physical activities, thus helping other industries by exposing these products to a wider audience.
* By promoting all of these industries on the software this then opens up the option for the software to take advantage of advertisements as an avenue of profiting from the platform.

## Technology Challenges

[Any technological constraints that the project will be under. Any new technologies you may need to use]

For the posting aspect of the prospective software a user may refresh the page so any new posts that may have been posted in that time will have to be shown, so one technical requirement may be to be able to live update the application when its running to ensure that the user is able to view the most up to date content.

## References

[Mention books, articles, web sites, worksheets, people who are sources of information about the application domain, etc. Use proper and complete reference notation. Give links to documents as appropriate. You should use the APA Documentation model (Alred, 2003, p. 144).]

# General Description

## Product Perspective

[Describe the context and origin of the product.]

Puzzle found its origins in a climber’s desire for a simpler way to track their climbing progress. The idea was originated by a climber for climbers.

## Product Features

[A high-level summary of the functions the software would perform and the features to be included.]

The product features include the ability for individual climbers and climbing gyms to create accounts and the ability for administrators to manipulate those accounts. Climbers can also add climbing routes to their profiles, where they can track their climbing progress, with different tracking options based on climbing style. For gyms, the functionality also includes the ability to create climbing routes. They can also create events with a title and information. For administrators, the functionality also includes the possibility to view and delete accounts.

## User Class and Characteristics

[A categorization and profiling of the users the software is intended for and their classification into different user classes]

Our website application does not expect our users to have any prior knowledge of a computer, apart from using a web browser, or knowledge of astronomy. Our website application has removed the need for them to have astronomy, math, or science knowledge and allows the user to focus on exploring the night sky.

## Operating Environment

[Specification of the environment the software is being designed to operate in.]

The application is designed to operate on the web across many different devices.

## Constraints

[Any limiting factors that would pose challenge to the development of the software. These include both design as well as implementation constraints.]

Due to the use of a 3d engine, we had to limit the web browsers supported. To limit user error when entering the user’s address, we implemented a drop-down AJAX country, state, and city selection.

## Assumptions and Dependencies

[A list of all assumptions that you have made regarding the software product and the environment along with any external dependencies which may affect the project]

The software will be dependent on Spring Web and Thymeleaf in order to create and execute the MVC architecture that will be developed within NetBeans. The application will also use the World Time API (http://worldtimeapi.org/) that will display the current date and time on the home dashboard for everyone to see.

# Functional Requirements

[Statements of services the system should provide, how the system should react to particular inputs and how the system should behave in particular situations.]

## Primary

[All the requirements within the system or sub-system in order to determine the output that the software is expected to give in relation to the given input. These consist of the design requirements, graphics requirements, operating system requirements and constraints if any.]

* FR0: The system will allow the user to lookup of vehicle owner information based on license plate number. This information will contain owner’s permit number, assigned lot, and previous violations including tow history.
* FR1: The system will allow the user to enter a new vehicle into the vehicle violation database.
* FR2: The system will allow the user to issue a ticket. The ticket information will be issued in electronic and paper form.

## Secondary

[Some functions that are used to support the primary requirements.]

* Password protection for information only accessible to employees, managers, and each individual table.
* Authorization scheme so that customers can only alter and see their orders and no other customers’ orders.

# Technical Requirements



## Operating System and Compatibility

[The environments that will be needed to operate the system]

The application will be compatible with any operating system that is able to view and to interact with traditional web pages.

## Interface Requirements

### User Interfaces

[The logic behind the interactions between the users and the software. This includes the sample screen layout, buttons and functions that would appear on every screen, messages to be displayed on each screen and the style guides to be used.]

### Hardware Interfaces

[All the hardware-software interactions with the list of supported devices on which the software is intended to run on, the network requirements along with the list of communication protocols to be used.]

The web application will run on any hardware device that has access to the internet, the ability to display webpages, and the ability to interact with web pages. This includes, but is not limited to, smartphones, tablets, desktop computers, and laptops.

### Communications Interfaces

[Determination of all the communication standards to be utilized by the software as a part of the project]

It must be able to connect to the internet as well as the local database on phpMyAdmin.

The communication protocol, HTTP, must be able to connect to the World Time API and return the current date and time.

### Software Interfaces

[The interaction of the software to be developed with other software components such as frontend and the backend framework to the used, the database management system and libraries describing the need and the purpose behind each of them.]

We will use React and Spring Boot ThymeLeaf to help build the frontend, as well as JPA for the backend database functionality. We will also use Spring Boot with Java to connect the frontend to the backend.

# Non-Functional Requirements

[Constraints on the services or functions offered by the system (e.g., timing constraints, constraints on the development process, standards, etc.). Often apply to the system as a whole rather than individual features or services.]

## Performance Requirements

[The performance requirements need to be specified for all the functional requirements.]

* NFR0(R): The local copy of the vehicle violation database will consume less than 20 MB of memory
* NFR1(R): The system (including the local copy of the vehicle violation database) will consume less than 50MB of memory
* NFR2(R): The novice user will be able to create and print a ticket in less than 5 minutes.
* NFR3(R): The expert user will be able to create and print a ticket in less than 1 minute.

## Safety Requirements

[List out any safeguards that need to be incorporated as a measure against any possible harm the use of the software application may cause.]

## Security Requirements

[Privacy and data protection regulations that need to be adhered to while designing of the product.]

* The software only asks for relevant user personal data, not excessive or unnecessary.
* The software is only able to be accessed by users with an account.
* The software only asks for the permissions that are required

## Software Quality Attributes

[Detailing on the additional qualities that need to be incorporated within the software like maintainability, adaptability, flexibility, usability, reliability, portability etc.]

### Availability

The software needs to be able to cope and deal with any possible errors/ flaws that it may encounter, so therefore error catching will need to be implemented into relevant sections as to ensure that an error does not break the software. Additionally, sufficient testing will need to take place as to identify areas where errors occur to then correct these.

### Correctness

The software will need to perform the exact tasks/ processes that we have defined in this document.

### Maintainability

The software will need to be able to updated and improved as time progresses, once the software is released then users may have ideas and suggestions to improve the product be this informing about bugs and problems or features that they want to have added to provide a better service. Therefore, the code of the software needs to be able to be added to easily in order for this to be a possibility.

### Reusability

Using code that is already available. Such as code libraries and APIs. Also by splitting the code created into different modules and sections as so that it can be used by multiple areas of the software, thus bringing down cost and time.

### Portability

The software will initially be created using a web platform and will be designed initially as a desktop product, the software will be responsive in order to fit to different sized screens in order to maximize the number of devices that it can be clearly viewed on. In the future in order to maximize portability the product will be adapted into a mobile application as this will allow for a larger number of users to use the platform.

## Process Requirements

### Development Process Used

[Software Process Model]

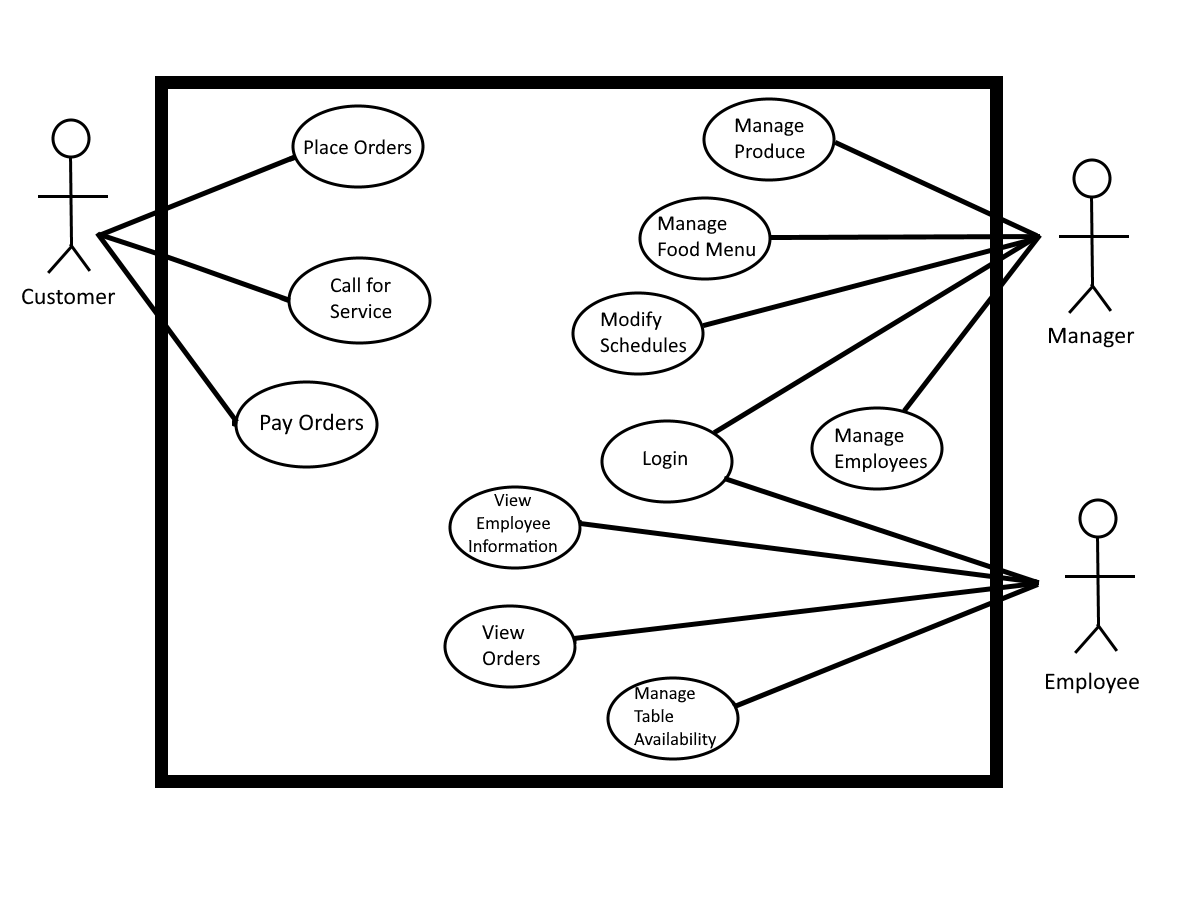
### Time Constraints

### Cost and Delivery Date

## Other Requirements

TBD

## Use-Case Model Diagram



## Use-Case Model Descriptions

### Actor: Actor Name (Responsible Team Member)

* **Use-Case Name**: [Brief Use-Case Description]
* **Use-Case Name**: [Brief Use-Case Description]

### Actor: Actor Name (Responsible Team Member)

* **Use-Case Name**: [Brief Use-Case Description]
* **Use-Case Name**: [Brief Use-Case Description]

### Actor: Actor Name (Responsible Team Member)

* **Use-Case Name**: [Brief Use-Case Description]
* **Use-Case Name**: [Brief Use-Case Description]

## Use-Case Model Scenarios

### Actor: Actor Name (Responsible Team Member)

* **Use-Case Name**:
  + **Initial Assumption**:
  + **Normal**:
  + **What Can Go Wrong**
  + **Other Activities**:
  + **System State on Completion**:
* **Use-Case Name**:
  + **Initial Assumption**:
  + **Normal**:
  + **What Can Go Wrong**:
  + **Other Activities**:
  + **System State on Completion**:

### Actor: Actor Name (Responsible Team Member)

* **Use-Case Name**:
  + **Initial Assumption**:
  + **Normal**:
  + **What Can Go Wrong**:
  + **Other Activities**:
  + **System State on Completion**:
* **Use-Case Name**:
  + **Initial Assumption**:
  + **Normal**:
  + **What Can Go Wrong**:
  + **Other Activities**:
  + **System State on Completion**:

### Actor: Actor Name (Responsible Team Member)

* **Use-Case Name**:
  + **Initial Assumption**:
  + **Normal**:
  + **What Can Go Wrong**:
  + **Other Activities**:
  + **System State on Completion**:
* **Use-Case Name**:
  + **Initial Assumption**:
  + **Normal**:
  + **What Can Go Wrong**:
  + **Other Activities**:
  + **System State on Completion**:

# Design Documents

## Software Architecture

## High-Level Database Schema

## Software Design

### State Machine Diagram: Actor Name (Responsible Team Member)

### State Machine Diagram: Actor Name (Responsible Team Member)

### State Machine Diagram: Actor Name (Responsible Team Member)

## UML Class Diagram

# Scenario

## Brief Written Scenario with Screenshots